# OpenAD 1/2

Jean Utke[1]

[1]University of Chicago and Argonne National Laboratory

ECCO Workshop
Nov. 9/10, 2009


UChicago ▶
Argonne LLC

# outline

- part 1: OpenAD
  - current
  - changes
  - research
  - some other apps
  - for MITgcm
- part 2: adjoint Experiments
  - with Patrick and Chris
  - MITgcm experiments

# why automatic differentiation?

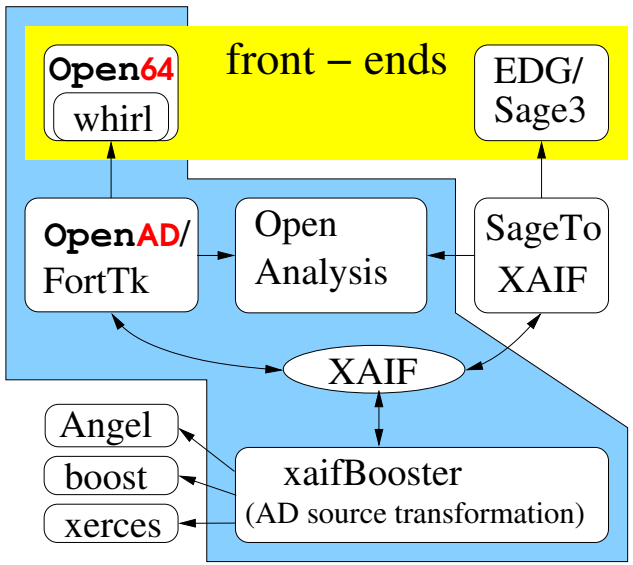
given: some numerical model $\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x}) : \mathbb{R}^n \mapsto \mathbb{R}^m$ implemented as a (large / volatile) program

wanted: sensitivity analysis, optimization, parameter (state) estimation, higher-order approximation...

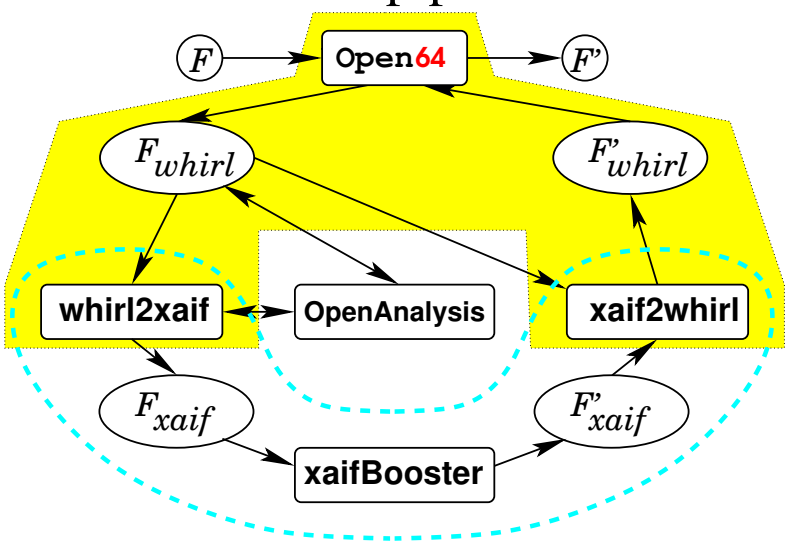
1. don't pretend we know nothing about the program
   (and take finite differences of an oracle)
2. get machine precision derivatives as $\boldsymbol{J}\dot{\boldsymbol{x}}$ or $\bar{\boldsymbol{y}}^T\boldsymbol{J}$ or ...
   (avoid approximation-versus-roundoff problem)
3. the reverse (aka adjoint) mode yields "cheap" gradients
4. if the program is large, so is the adjoint program, and
   so is the effort to do it manually ... easy to get wrong but hard to debug

# OpenAD overview - current

- www.mcs.anl.gov/OpenAD
- forward and **reverse**
- source transformation
- modular design
- aims at large problems
- language independent transformation
- researching combinatorial problems
- current Fortran front-end Open64 (Open64/SL branch at Rice U)
- uses *association by address* (i.e. has an active type)
- Rapsodia for higher-order derivatives via type change transformation

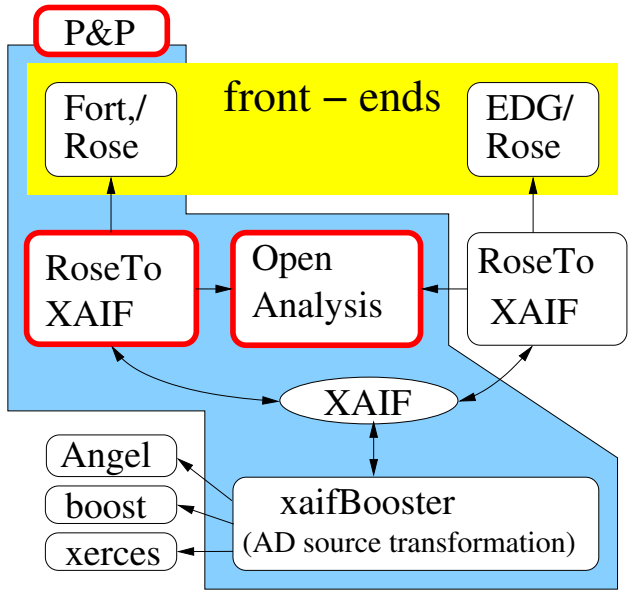



Fortran pipeline:

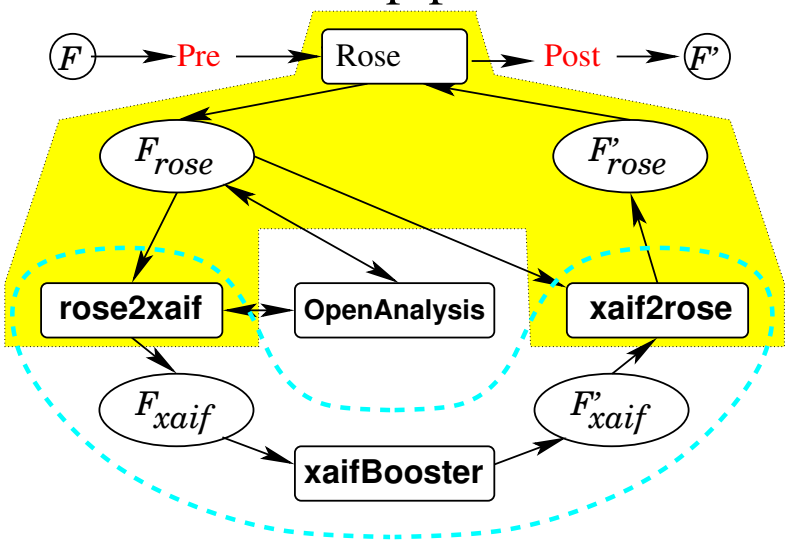


# OpenAD overview - changes

- expanded language coverage (common blocks, equivalence, unstrucctured control flow, intrinsics,...)
- new pre- and postprocessor (python, MITgcm consequences)
- migration from Open64 to Rose (LLNL)




Fortran pipeline:



# some research toopis

- adjoinable MPI
- optimal local preaccumulation (scarcity)
- additional parallelism from checkpointing
- higher order derivatives (in parallel)
- ...

# some research toopis

- adjoinable MPI
- optimal local preaccumulation (scarcity)
- additional parallelism from checkpointing
- higher order derivatives (in parallel)
- ...
- make it work on code *<insert something here>* ...

# some other applications

- suite of reactor models
  - old style Fortran
  - equivalence, unstructured control flow,...
- transport of nuclear materials (container safety)
  - Fortran 9X
  - dependecies via files
  - dynamic memory
- forthcoming: ice sheet models (NSF and DOE projects)

needs migration to Rose

# for MITgcm

- installed on beagle (updated/recompiled nightly)
- w. Chris (use w/o intervention)
  - cost function change,
  - adding extra output
  - compiler optimization
  - computational cost
- w. Patrick 20 year 1x1 run on beagle
  - setup hurdle (find the right combination of modules for the sge run script)
  - bottleneck checkpointing via NFS (switch to local disk)
- usability: remove extra steps e.g. Common Block to Module conversion, some specific changes to non-transformed files. e.g. cost_final
- next step w. Chris: “high-res” run